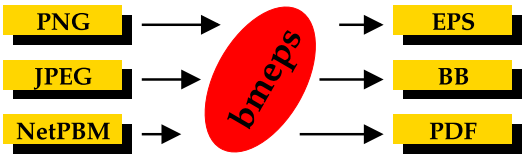

Bmeps 2.0.8 - The manual

Dipl.-Ing. D. Krause

January 4, 2010

Contents

1 Overview

1.1 Program purpose

The "bmeps" program converts PNG-, JPEG- and NETPBM to EPS, PDF and BB files (bounding box files for LaTeX).

Additionally bmeps can process TIFF files which can be read using the *TIFFReadRGBAImage()* from the libtiff library.

The package also builds a library libbmeps which can be linked into dvips.

1.2 License terms

1.2.1 License terms

The software is licensed to you under the terms of a BSD-style license:

- Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
  - Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  - Redistributions in binary form must reproduce the above opyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  - Neither the name of the Dirk Krause nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
- This software is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed.
  In no event shall the copyright owner or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

1.2.2 Exceptions

- dvips-mods subdirectory
  The files in the dvips-mods subdirectory are derived from the original dvips sources and remain under the same license conditions and copyright statements as the original sources. The current version of dvips – the version you should use to build a modified dvips – can be found in the teTeX distribution[1], the file is tetex-src.tar.gz. After unpacking the archive you will find a directory tetex-src-2.0.2/texk/dvipsk (the version number may vary), see the sources and documentation in this directory for copyright and license details.

- contrib/kant_krishna directory and DOCU/kant_krishna/excel_to_eps.txt file
  The file excel_to_eps.vb was provided by Krishna Kant. No copyright and/or license terms were specified for this file.
  You should leave the Author:...-line intact when distributing this file.

[1] ftp://ftp.dante.de/tex-archive/systems/unix/teTeX/current/distrib

2 Installation

The following libraries are needed to build bmeps from source:

- zlib (required)
- libbz2 (optional)
- libpng (required)
- jpeglib (required)
- NetPBM (optional)
- libtiff (optional)
- dklibs (required)

After unpacking the source archiv run

```
./configure
make
make install
```

to build and install bmeps.

3 Usage

3.1 Configuration mechanism

Bmeps provides a large number of configurable settings. To avoid typing all settings on the command line each time the program is run, one can create a configuration file and choose one configuration from the file using the "-l..." command line option.

The "-o..." command line option can be used to overwrite some settings of the configuration choosen by -l.

If the "-l" option is omitted bmeps uses the following tests to find a configuration (the search is aborted after the first test succeeds):

- The output file name – if specified – is inspected whether it ends on ".eps", ".ps", ".pdf" or ".bb". In these cases the configuration "eps", "ps", "pdf" or "bb" is used.
- The EPSOUTPUT environment variable – if defined – contains the name of the configuration to use.
- If a default configuration was previously set using

  ```
  bmeps -c -l ...
  ```

  this configuration is used.
- "eps" is used if no configuration was found yet.

A number of configurations is compiled into bmeps, no configuration file is necessary to use these configurations, see table 1.

Table 1: Basic configurations

Name	Meaning
ps1	Create PS level 1 output.
ps2	Create PS level 2 output.
ps3	Create PS level 3 output.
ps	Synonym for ps2.
eps1	Create EPS level 1 output.
eps2	Create EPS level 2 output.
eps3	Create EPS level 3 output.
	to be continued...

	Continuation
eps	Synonym for eps2.
faxps	Create a printable file (PS level 2) on A4 paper.
pdf12	Create PDF 1.2 output-
pdf13	Create PDF 1.3 output.
pdf14	Create PDF 1.4 output.
pdf	Synonym for pdf14
faxpdf	Create grayscaled PDF-1.4 output, use A4 paper.
bb	Create a bounding box for LaTeX.

3.2 Command line options

Type

`bmeps [`*options*`] [`*input-file*`] [`*output-file*`] [`*options*`]`

or

`bmeps [`*options*`] [ `*directory*`]`

to run bmeps.
The following command line options are available:

- `-h`
  shows a help text.
- `-v`
  shows the version number and the license conditions.
- `-c -l...`
  save the "-l" settings as permanent options (defaults).
- `-u`
  removes the permanent options.
- `-C`
  shows the current configuration.
- `-l`*configuration*
  chooses a configuration, optional configuration entry overwrites may follow, separated by comma.
- `-o`*name=value*
  overwrites a configuration entry (can be used multiple times).
- `-f`*frame-number*
  `-f`*start-frame-end-frame*
  chooses the frames to process if an input file contains multiple frames (i.e. TIFF files).
- `-m`
  `-m-`
  activates/deactivates make mode. If bmeps is run on a directory, the directory contents is inspected and a conversion is started for each input file bmeps can handle.
  In make mode bmeps checks whether or not an up-to-date conversion output already exists. A conversion is started only if the output file is not yet present or not up to date.

- `-a`
  automatically chooses an output file name if necessary. The output file name is based on the input file name and the output driver.
  You can use

  ```
  bmeps -a -lbb picture.png
  ```

  to produce the picture.bb file.

- `-t`*file-type*
  specifies the file type ("png" for PNG files, "jpg" for JPEG files, "pbm" for NetPBM files or "tiff" for TIFF files). This option is only necessary if bmeps processes standard input and no input file name is known.

- `-A`
  `-A-`
  suppresses/enables the notification about alpha channels transferred from PNG or TIFF to PDF. If PDF files containing alpha channels are used by LaTeX sources the preamble should contain a special setup instruction to prevent PDF viewers from using CMYK as intermediate color space. So when creating such PDF files bmeps issues a reminder.

If the "-l" option is not used bmeps attempts to find a configuration name by inspecting the output file name. In this case all options to overwrite configuration entries *must* be placed after the name of the output file.

3.3 Configuration file - general structure

The configuration file consists of one or multiple section, each of them describing one configuration.

The section is started by the configuration name in square brackets, i.e.

```
[myprinter]
```

An optional parent configuration to inherit from can be specified separated by a colon:

```
[myprinter:ps2]
```

The section body contains lines consisting of one configuration entry each. For a list of configuration entry names see section 3.4 on the next page.

An example configuration for a PS2 grayscaled printer might look like this:

```
[myprinter:ps2]
color = no
```

3.4 Configuration file entries

Configuration entries can be specified using different ways. To create A4 output for a grayscaled laser printer you can use one of the following commands:

```
bmeps -l ps2 -o color=no -o media.size=A4 ...
bmeps -l ps2 -o c=n -o m.s=A4
bmeps -l ps2,color=no,media.size=A4 ...
bmeps -l ps2,c=n,m.s=A4 ...
```

Alternatively you can create a configuration file

```
[printer:ps2]
color           =       no
media size      =       A4
```

and use the "-l" option to choose the configuration:

```
bmeps -l printer ...
```

I recommend to use the full configuration entry name in the configuration file. Abbreviations should be used on the command line only.

The following configuration entries can be used:

- `output type` = *string*
  `o.t`
  Output type, either "ps", "pdf" or "bb" (bounding box file for LaTeX).

eps1:	ps
eps/eps2:	ps
eps3:	ps
ps1:	ps
ps/ps2:	ps
ps3:	ps
pdf12:	pdf
pdf13:	pdf
pdf/pdf14:	pdf
faxpdf:	pdf
bb:	bb

- `level` = *string*
  `l`
  Format level, either "1", "2" or "3" for PS/EPS or "1.2", "1.3" or "1.4" for PDF.

eps1: 1
eps/eps2: 2
eps3: 3
ps1: 1
ps/ps2: 2
ps3: 3
pdf12: 1.2
pdf13: 1.3
pdf/pdf14: 1.4
faxpdf: 1.4

- `color` = *boolean*
  `c`
  decides whether to create colored or grayscaled output.
  eps1: off
  eps/eps2: on
  eps3: on
  ps1: off
  ps/ps2: on
  ps3: on
  pdf12: on
  pdf13: on
  pdf/pdf14: on
  faxpdf: off
- `pdf fit` = *string*
  `p.f`
  establishes the initial zoom factor when opening a PDF file:
  - `width`
    image width fits window width,
  - `height`
    image height fits window height,
  - `page`
    shows page completely in window.

  pdf12: page
  pdf13: page
  pdf/pdf14: page
  faxpdf: width
- `interpolate` = *boolean*
  `i`

enables/disables image interpolation in PDF and PS level 2 and 3 output.

eps/eps2:	on
eps3:	on
ps/ps2:	on
ps3:	on
pdf12:	on
pdf13:	on
pdf/pdf14:	on
faxpdf:	on

- `jpeg interpolate` = *boolean*
  `j.i`
  works in conjunction with "interpolate" and enables/disables image interpolation when passing through DCT encoded data from a JPEG file. If this option is disabled, interpolation is turned of for DCT passthrough.

eps/eps2:	off
eps3:	off
ps/ps2:	off
ps3:	off
pdf12:	off
pdf13:	off
pdf/pdf14:	off
faxpdf:	off

- `encoding` = *string*
  `e`
  lists the encoding and compression mechanisms to use. List items are separated by a colon, abbreviations can be used, i.e. "a:r:f:d" for "ascii85:run-length:flate:dct".
  The order of appearance in the list is insignificant, see figure 1 on the following page for possible sequences of encodings. The following encodings/-compressions can be used:
  - `ascii85`
    `a`
    ASCII85-Encoding. If deactivated, ASCII-Hex-Encoding is used.
  - `run-length`
    `r`
    Run-length compression.
  - `flate`
    `f`
    Flate compression.

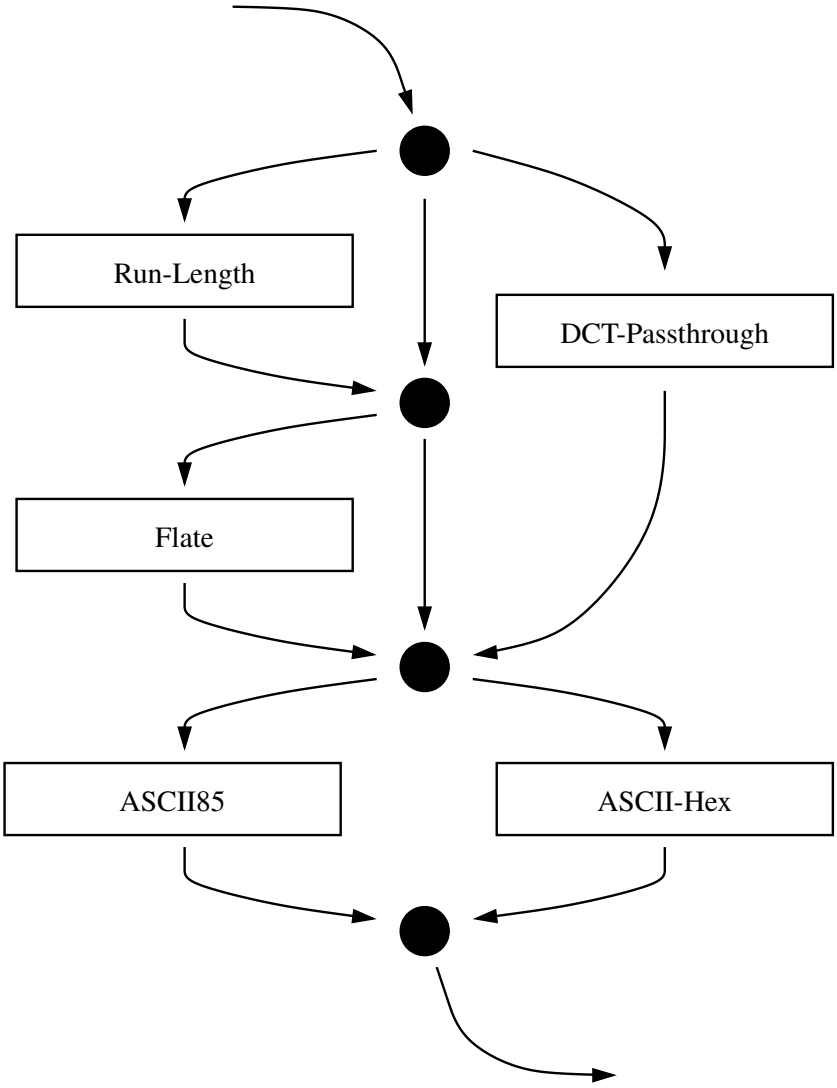

Figure 1: Compression and encoding algorithms

– `dct`
`d`
DCT compression.
Bmeps does not DCT compress data, but it can pass through DCT compressed data from JPEG files.

eps1:	ASCII-Hex
eps/eps2:	ASCII85, DCT
eps3:	ASCII85, flate, DCT
ps1:	ASCII-Hex
ps/ps2:	ASCII85, DCT
ps3:	ASCII85, flate, DCT
pdf12:	ASCII85, flate, DCT
pdf13:	ASCII85, flate, DCT
pdf/pdf14:	ASCII85, flate, DCT
faxpdf:	ASCII85, flate, DCT

- `jfif sof set` = *string*
  `j.s.s`
  PostScript level 2 only allows SOF_0 and SOF_1 markers in DCT compressed data.
  Some printers and interpreters implement support for additional markers. To specifiy the set of marker, create a colon-separated list of number, i.e. "0:1:2" to allow SOF_0, SOF_1 and SOF_2.
  eps/eps2: 0:1
  eps3: 0:1
  ps/ps2: 0:1
  ps3: 0:1
  pdf12: 0:1
  pdf13: 0:1
  pdf/pdf14: 0:1
  faxpdf: 0:1

- `predictor` = *string*
  chooses a predictor to improve the flate compression when producing PDF output. Choose one from "none" (no prediction, default), "tiff", "png-sub", "png-up", "png-average" or "png-paeth".
  The "tiff" predictor can be used with any color depth, the "png-*" predictors can be used with 8 or 16 bits per component only.
  For "normal" PNG images (i.e. screenshots, logos, diagrams...) there is no need to use a predictor, only for special cases (i.e. images with color gradients, photos saved as PNG...). It depends on the image which predictor will produce the smallest output.

- `transfer alpha channel` = *string*
  `t.a.c`
  transfers alpha channel data to output (only PDF 1.4). This option deactivates mixing against a background and creation of image masks.
  pdf/pdf14: on
  faxpdf: on

- `allow pdf page attributes`
  `a.p.p.a`
  allows/denies the creation of PDF page attributes.
  PDF page attributes are needed when transferring an alpha channel to PDF output.

- `mix` = *boolean*
  `m`

activates mixing against a background color if the input images contains alpha data. If available bmeps uses the background color specified in the input file, a default background is used otherwise.

eps1:	on
eps/eps2:	on
eps3:	on
ps1:	on
ps/ps2:	on
ps3:	on
pdf12:	on
pdf13:	on
pdf/pdf14:	off
faxpdf:	off

- `background` = *rgb-triple*
  `b`
  specifies the default background for mixing, the colors are given as numbers in the range from 0.0 to 1.0 separated by colon (i.e. "1:0.5:1" for magenta).

eps1:	white
eps/eps2:	white
eps3:	white
ps1:	white
ps/ps2:	white
ps3:	white
pdf12:	white
pdf13:	white
pdf/pdf14:	white
faxpdf:	white

- `always use default background` = *boolean*
  `a.u.d.b`
  always uses the default background color for mixing, no matter whether or not the input contains a background color chunk.

eps1:	off
eps/eps2:	off
eps3:	off
ps1:	off
ps/ps2:	off
ps3:	off
pdf12:	off
pdf13:	off
pdf/pdf14:	off
faxpdf:	off

- `create image mask` = *boolean*
  `c.i.m`
  converts input alpha channel into an image mask.

eps3:	off
ps3:	off
pdf12:	off
pdf13:	off
pdf/pdf14:	off
faxpdf:	off

- `image mask trigger level` = *numeric*
  `i.m.t.l`
  specifies the trigger level for conversion of alpha data to image masks, must be in the range $0 \leq x \leq 1$.

eps3:	0.000001
ps3:	0.000001
pdf12:	0.000001
pdf13:	0.000001
pdf/pdf14:	0.000001
faxpdf:	0.000001

- `use resolution chunk` = *boolean*
  `u.r.c`
  attempts to use resolution data from input to scale the output image.

eps1:	off
eps/eps2:	off
eps3:	off
ps1:	off
ps/ps2:	off
ps3:	off
pdf12:	off
pdf13:	off
pdf/pdf14:	off
faxpdf:	off

- `media size` = *string*
  `m.s`
  specifies a media size to fit output:
  - media size name: A3, A4, A5, B4, B5, Letter, Legal, Tabloid, Ledger, Statement, Executive, Folio, Quarto or 10x14.
  - 4 numbers separated by colon: x0, y0, x1 and y1.
  - 8 numbers separated by colon: bx0, by0, bx1, by1, ix0, iy0, ix1 and iy1. The b... values specify the bounding box written to output, the image is placed in the rectangle specified by the i... values. The b...-rectangle must completely include the i...-rectangle.

  Note: "u.r.c" und "m.s" can not be combined.

eps1:	-
eps/eps2:	-
eps3:	-
ps1:	-
ps/ps2:	-
ps3:	-
pdf12:	-
pdf13:	-
pdf/pdf14:	-
faxpdf:	A4
bb:	-

- `separated dictionary` = *boolean*
  `s.d`
  uses a "...dict begin ... end" construct to avoid overwriting existing definitions.

eps/eps2: on
eps3: on
ps/ps2: on
ps3: on

- `vmreclaim` = *boolean*
  `v`
  adds "1 vmreclaim" at the end of output to suggest a garbage collection. This options requires the separated dictionary ("s.d").
  eps/eps2: off
  eps3: off
  ps/ps2: off
  ps3: off

- `operator dictionary` = *boolean*
  `o.d`
  provides a dictionary as argument to the image operator (PS level 2, or 3).
  eps/eps2: on
  eps3: on
  ps/ps2: on
  ps3: on

- `multiple data sources` = *boolean*
  `m.d.s`
  creates multiple data sources for red, green and blue. This allows run-length compression for lines and areas in the same color.
  eps/eps2: on
  eps3: off
  ps/ps2: on
  ps3: off

- `showpage` = *boolean*
  `sh`
  activates/deactivates the use of the showpage operator in PS output. Output images for stand-alone viewing should contain a showpage operator. Images for inclusion into other documents (i.e. using LaTeX) should not contain it.
  Note: If the input contains multiple frames, the showpage operator is always used.

eps1:	off
eps/eps2:	off
eps3:	off
ps1:	on
ps/ps2:	on
ps3:	on

- `dsc comments` = *boolean*
  `dsc.c`
  create DSC comments in PS/EPS output.

eps1:	on
eps/eps2:	on
eps3:	on
ps1:	off
ps/ps2:	off
ps3:	off

- `draft` = *boolean*
  `d`
  activates draft mode. In draft mode bmeps only reads the image size and draws a placeholder rectangle.

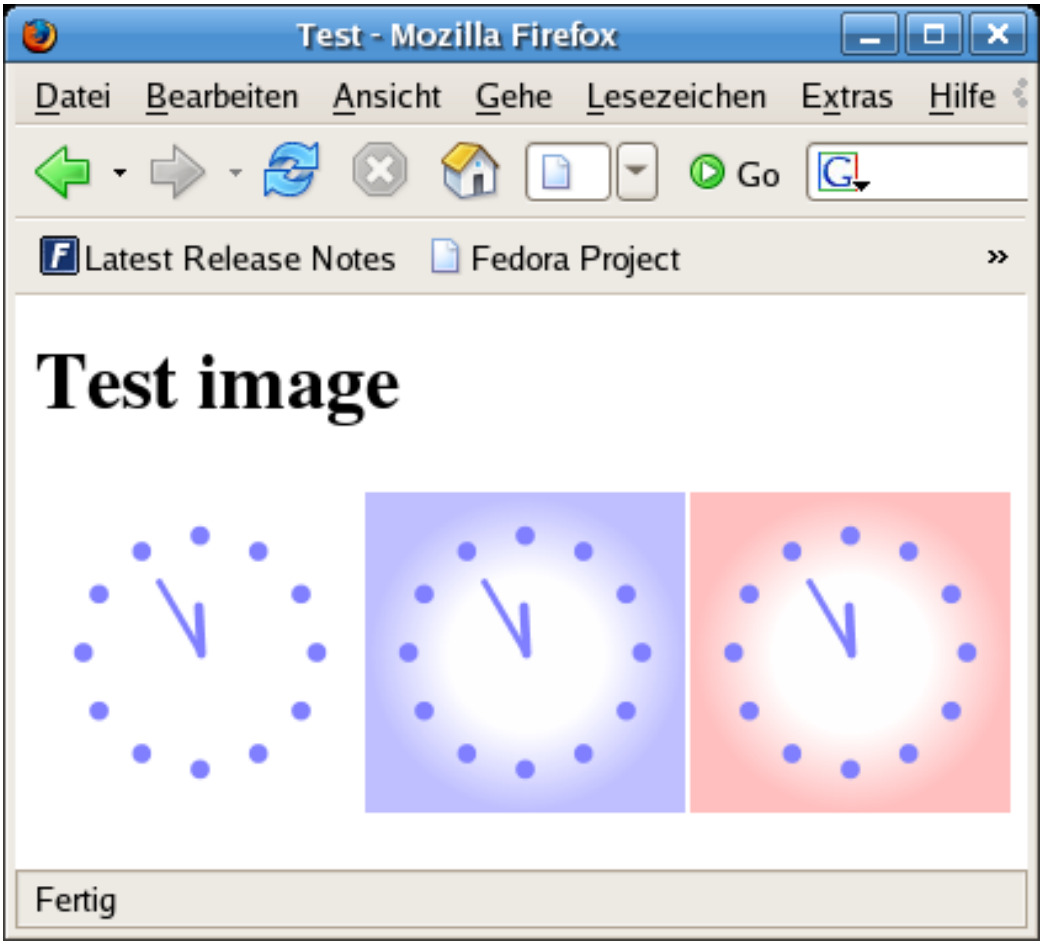

Figure 2: Example PNG file, different background colors.

3.5 LaTeX and bmeps

3.5.1 Examples

Test image For demonstation purposes we use a self-made file fbt.png containing a clock showing five before twelve.
The alpha channel contains a radial alpha gradient. Figure 2 shows the image in front of different backgrounds.

LaTeX document We use a beamer presentation containing just one foil.

```
\documentclass{beamer}
\mode<presentation>{\usetheme{Madrid}
\setbeamercovered{transparent}}
\usepackage[german]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{ifpdf}
\usepackage{graphicx}
\usepackage{color}
\ifpdf\hypersetup{pdfpagemode=FullScreen}\fi
\title[Beamer and bmeps]{Using bmeps with the beamer class}
\author[Krause]{D.~Krause}
\subject{bmeps}
\begin{document}
% \beamertemplateshadingbackground{yellow!50}{blue!50}
\begin{frame}
\frametitle{Image over colored background}
\includegraphics[width=5cm]{fbt1}
\end{frame}
\end{document}
```

In the beginning we use a white background, later we use yellow background and a blue-to-yellow transition.

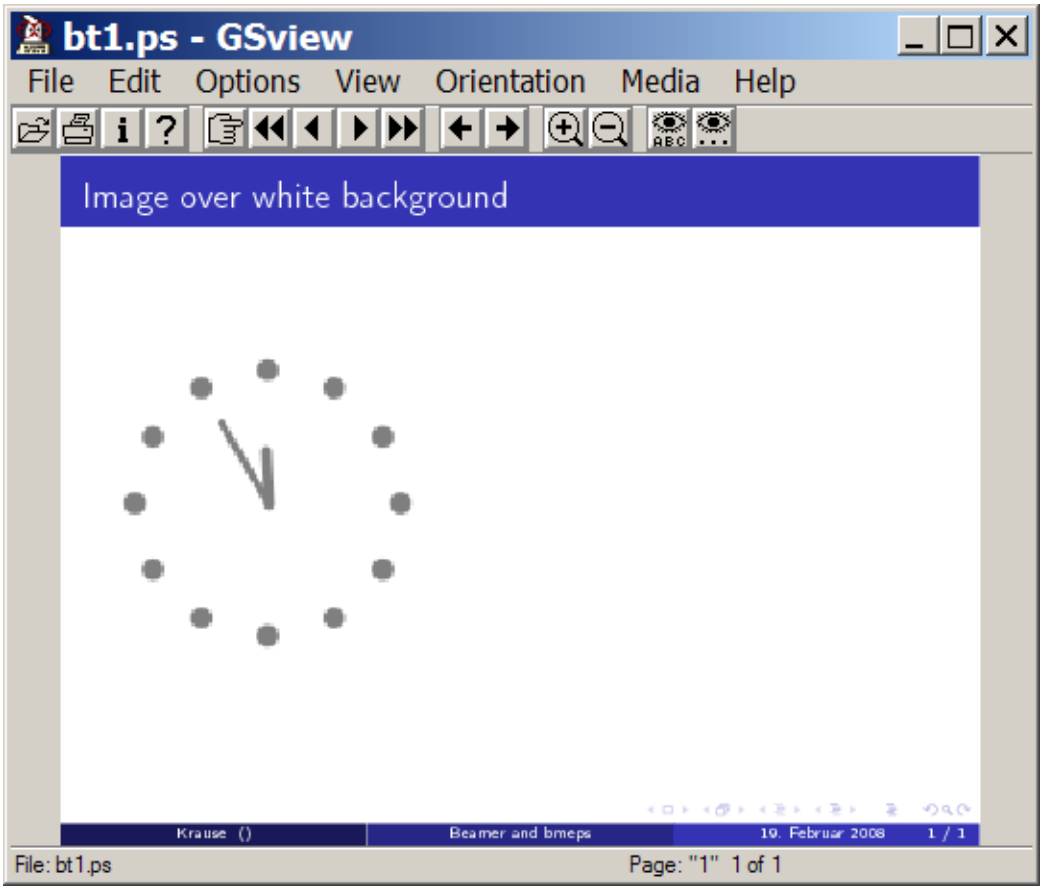

Figure 3: Example 1

PS-level 1 The image is converted to EPS using

```
bmeps -leps1 fbt.png fbt1.eps
```

the presentation is build by

```
latex bt1 && latex bt1 && dvips bt1
```

This produces a result as in figure 3.

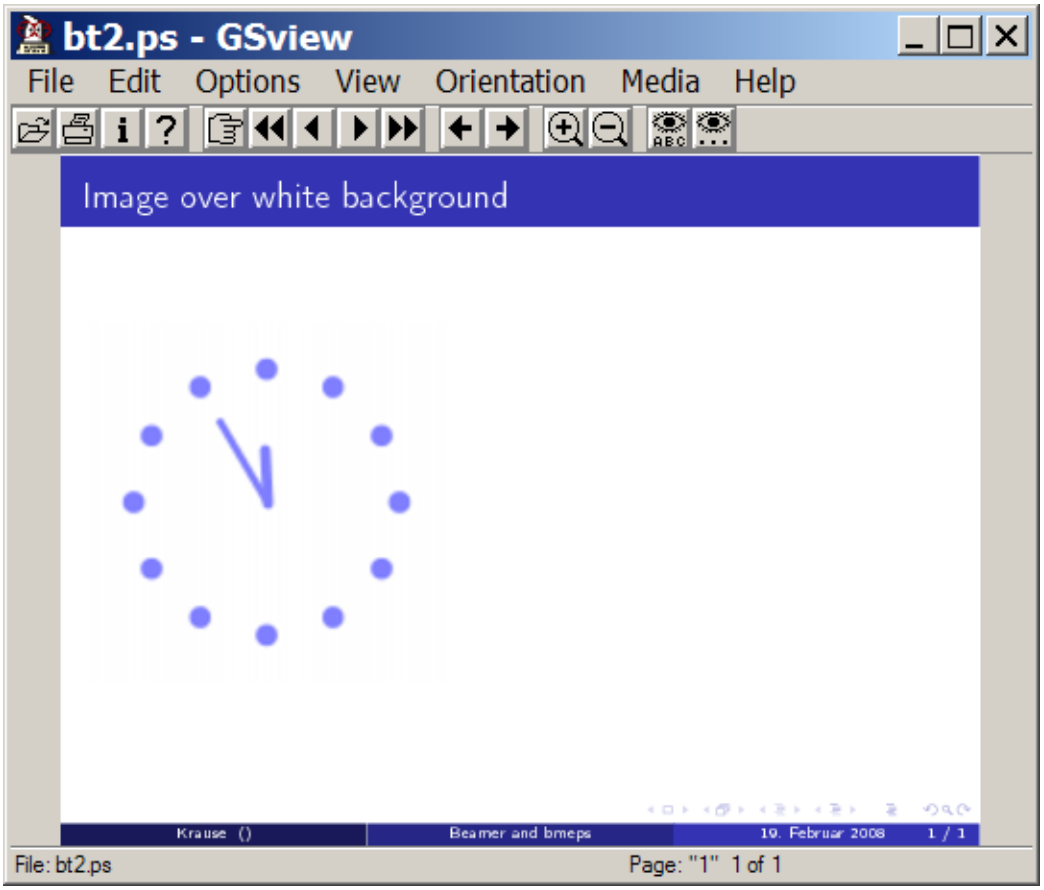

Figure 4: Example 2

PS-level 2 The conversion command

```
bmeps -leps2 fbt.png fbt2.eps
```

produces output as in figure 4 (fbt2.eps is included in the presentation now).

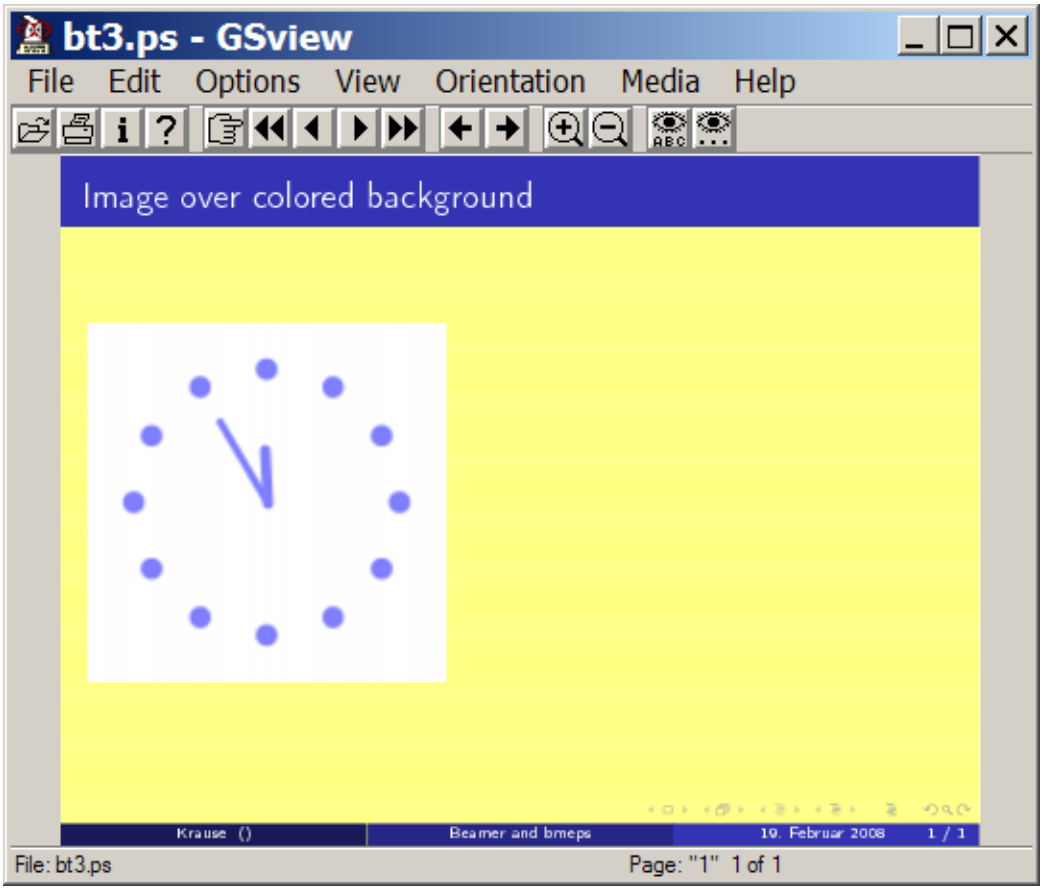

Figure 5: Example 3

Colored background If we use a colored background in the presentation the clock still has a white background because the white color is contained in the EPS file, see figure 5.

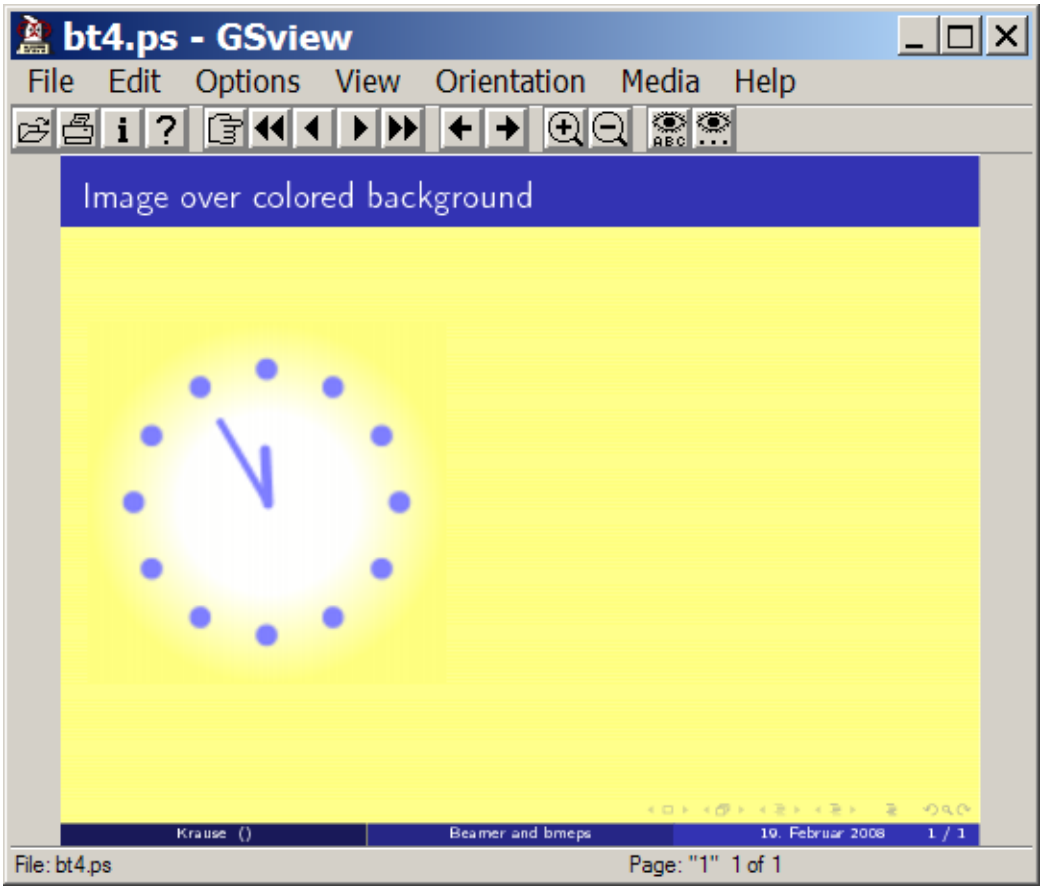

Figure 6: Example 4

Better background The yellow backgound in the presentation uses RGB 0xFF, 0xFF 0x7F. This results in 1, 1, 0.5 when converted into the range $[0;1]$. So we use the image conversion command

```
bmeps -leps2,m=y,b=1.0:1.0:0.5,a.u.d.b=y fbt.png fbt4.eps
```

The command mixes against a background color (m=y), the default background color is 1.0:1.0:0.5 and the default background color is always used for mixing (a.u.d.b=y) no matter whether or not the input contains a background chunk. Output can be seen in figure 6.

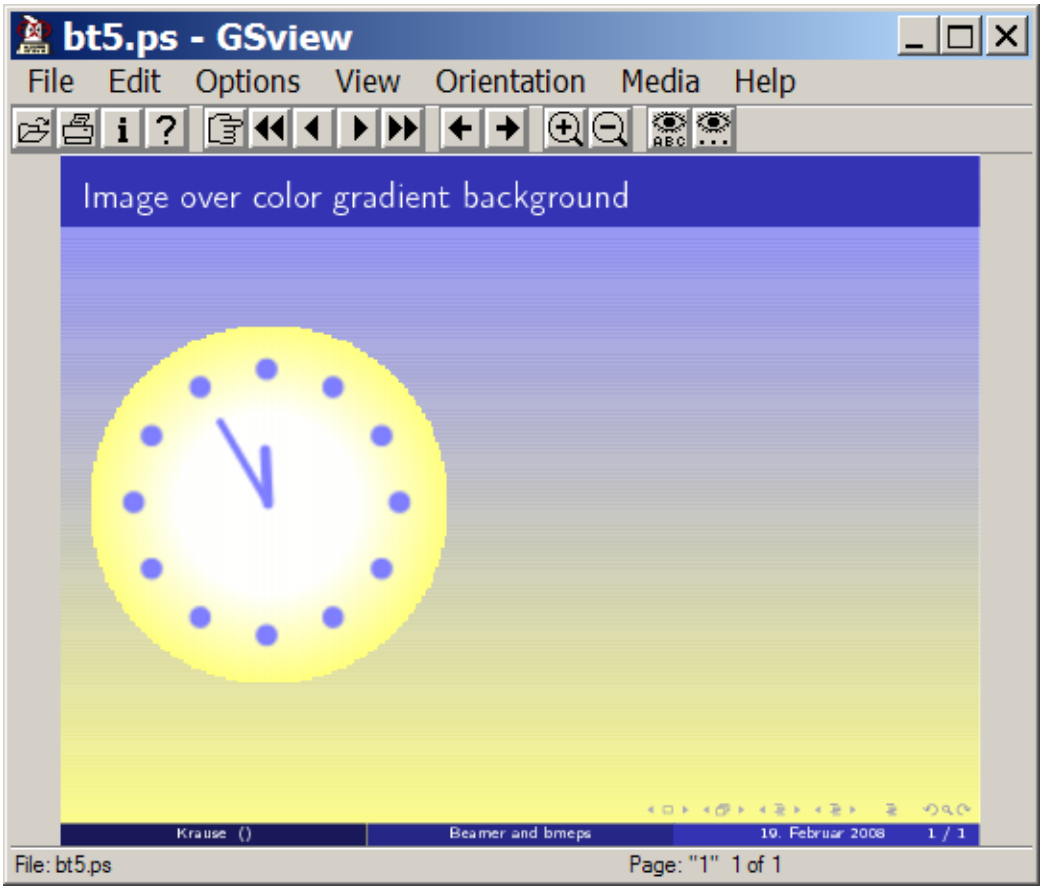

Figure 7: Example 5

Image mask (PS-level 3) The presentation in figure 7 uses a blue-to-yellow transition in the background. The clock image uses an image mask. Only pixels for which *opacity* > *triggerlevel* are drawn. The command to create fbt4.eps is:

```
bmeps -leps3,m=y,b=1.0:1.0:0.5,a.u.d.b=y,c.i.m=y fbt.png fbt5.eps
```

Alpha channel transfer (PDF) If we want partial transparency we need to transfer alpha channel data from input to output. This is only possible when creating PDF output. Figure 8 on the next page was produced by the commands

```
bmeps -lpdf fbt.png fbt7.pdf
pdflatex bt7 && pdflatex bt7 && pdflatex bt7
```

If a *.tex file references images containing alpha data an additional line is needed in the document preamble.

```
\documentclass{beamer}
\mode<presentation>{\usetheme{Madrid}
\setbeamercovered{transparent}}
\usepackage[german]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{ifpdf}
\usepackage{graphicx}
\usepackage{color}
\title[Beamer and bmeps]{Using bmeps with the beamer class}
\author[Krause]{D.~Krause}
\subject{bmeps}
% ----- The next line is important when dealing with alpha channels
\pdfpageattr {/Group << /S /Transparency /I true /CS /DeviceRGB>>}
\begin{document}
\beamertemplateshadingbackground{yellow!50}{blue!50}
\begin{frame}
\frametitle{Image over color gradient background}
\includegraphics[width=5cm]{fbt7.pdf}
\end{frame}
\end{document}
```

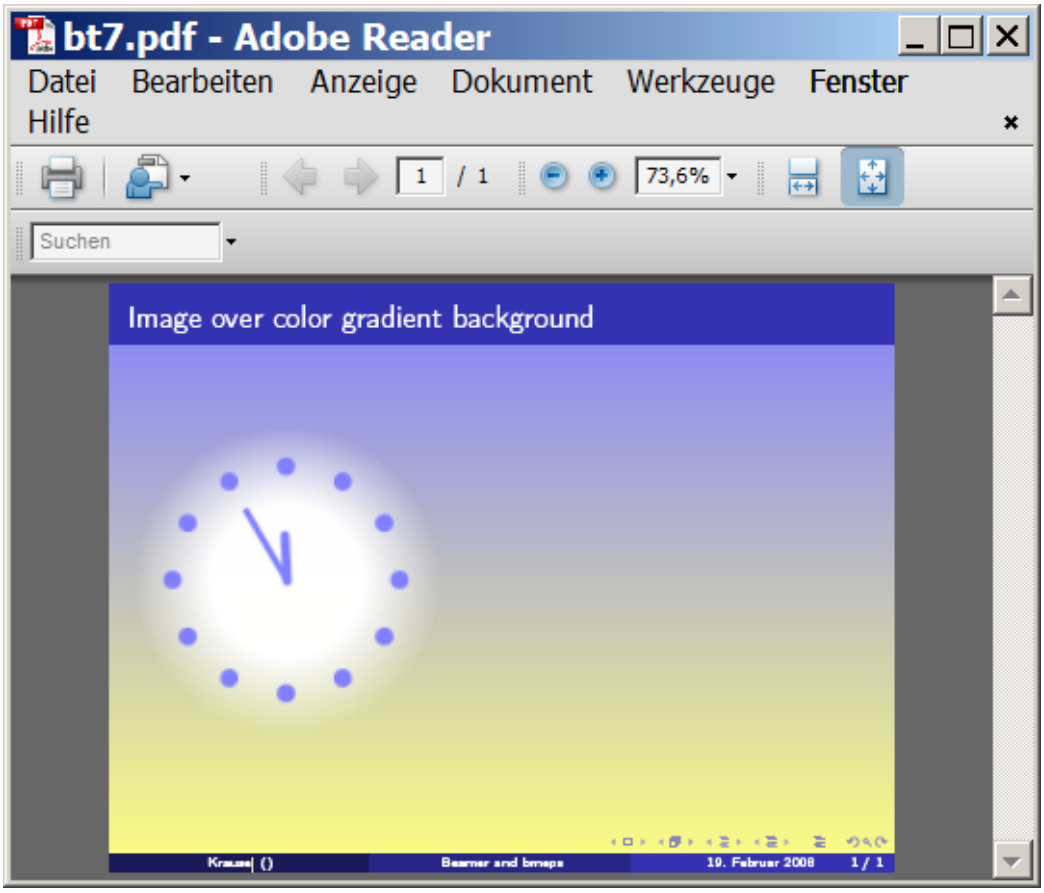

Figure 8: Example 6 (file sc7)

4 Further information on processing of different bitmap types

4.1 PNG

PNG is the preferred input format because

- PNG uses losless compression and a good compression ratio and
- there is a library (PNG reference library) capable to open/read all PNG images.

The PNG file format is a good choice for "artificial" images, i.e. logos, screenshots...

4.2 JPEG

The JPEG file format uses DCT compression, a lossy compression algorithm with variable compression rate.

It can be used for photos. It should not be used for logos, screenshots... or other images have large areas at the same color or sharp edges.

Some JPEG files can be used directly as DCT encoded data, bmeps only applies an additional ASCII-Hex- or ASCII85-encoding. The following conditions must be fulfilled to do so:

- The analysis function built into bmeps properly recognizes the image.
- The JPEG file contains exactly one begin-image instruction.
- The JPEG file contains only allowed SOF_n-markers.
- No conversion from colored input to grayscaled output is necessary.

If the test for these conditions fails, bmeps uses the jpeglib library to decode the image and uses the combination of run-length, flate and ASCII-Hex/ASCII85 to encode output.

4.3 NetPBM

The NetPBM tools allow to convert a wide variety of bitmap image types to NetPBM and from NetPBM to other image types. The NetPBM format is often used in temporary conversion steps.

To combine the NetPBM tools with bmeps use commands like:

```
xxxtopnm input.xxx | bmeps -lpdf -tpnm > output.pdf
```

4.4 TIFF

TIFF is not just one file format, it is a collection of different compression and encoding mechanisms.

There are standards, i.e. "TIFF V6.0 baseline" specifying collections of TIFF tags.

Multiple extension tags were created by several software vendors. It is not unusual for TIFF-processing programs to complain about unknown tags in input files.

Bmeps uses the *TIFFReadRGBAImage()* from libtiff to read TIFF images, so bmeps can only handle TIFF images fulfilling the following conditions:

- The file must follow the "TIFF V6.0 baseline" standard.
- The number of bits per component must be ≤ 8.
- The memory need for the image is $4 \cdot w \cdot h$ bytes. The system must be able to allocate this memory in one piece.

TIFF support was added to bmeps to convert files created by a fax gateway to PDF. This was tested successfully.

For other TIFF images bmeps will possibly fail. This is not a bug.

If your image editing or image creation software allows you to choose a file type, choose PNG if available, not TIFF. Or choose any other file type which can be converted to PNG or NetPBM.

5 New in version 2.0.0

5.1 More robust PS/EPS output

The output produced by previous versions of bmeps followed a scheme like

```
/pstr 128 string def
/inputf currentfile /ASCII85Decode
filter /RunLengthDecode filter def
gsave
0 128 translate 128 128 scale
128 128 8 [128 0 0 -128 0 0] { inputf pstr readstring pop }
image
K)^H&K)^H&K)^H&K)^H&K)^H&K)^H&\,QR/mHa_*rrq/DNK'1UM#RM<Nr4k:
% .... weitere codierte Daten
K)'=[
~>
grestore
currentdict /inputf undef currentdict /pstr undef
```

Some PS instructions are placed behind the encoded image data. If there is a problem during decoding the PS interpreter continues to scan for instructions at the file position where image reading aborted. No usefull PS instructions will be found, so the PS interpreter will stop processing and issues an error message.

The “filter” instructions open files which are never closed (at least not explicitly closed).

I compared bmeps output against the output of other programs producing EPS, especially jpeg2ps. Bmeps 2.0.0 now uses a scheme like:

```
{
gsave
13 dict begin
/fa currentfile /ASCII85Decode filter def
/fb fa /FlateDecode filter def
/sr 128 string def /sg 128 string def /sb 128 string def
/DeviceRGB setcolorspace
0 128 translate 128 128 scale
<<
/ImageType 1
/Width 128 /Height 128 /ImageMatrix [128 0 0 -128 0 0]
/MultipleDataSources true
/DataSource [
{ fb sr readstring pop }
{ fb sg readstring pop }
{ fb sb readstring pop }
]
/BitsPerComponent 8
/Decode [0 1 0 1 0 1]
/Interpolate true
>>
image
fb closefile
fa flushfile fa closefile
end
grestore
} exec
J3Vsg3$]7K#D>EP:q1$o*=mro@So+\<\5,H7Uo<*jE<[.O@Wn[3@'nb-^757
?K;B=Q#WO;koDO!%M.<jFY8C$]2rn<?/*&:mfh[_=,uiX>KqH^7HqKeXM^,2
~>
```

Using "{ ... } exec" the PS interpreter first reads the entire instruction set. Cleanup (closing files, restoring the graphics state) is done, no matter whether or not the image data was decoded successfully or not. All files are closed, the "flushfile" instructions "eats up" the data on the level of the lowest filter file object until the EOD marker "~>" is found.

5.2 Multiple data sources

Version 2.0.0 provides support for multiple data sources in EPS output.
If the input contains a red area the interleaved data source would contain bytes 0xFF, 0x00, 0x00, 0xFF, 0x00, 0x00, 0xFF, 0x00, 0x00... No run-length compression would be possible. Multiple data sources would contain 0xFF, 0xFF, 0xFF..., 0x00, 0x00, 0x00... and 0x00, 0x00, 0x00... Run-length compression is possible now.

The PS/EPS output contains image data line by line, for each line there is one string for the red-values, one string for the green-values, one for the blue-values. The image operator uses functions filling one string with data and placing this string on the stack. The string size is restricted to 16384 Bytes.
Summary: Multiple data sources can be used if the sequence of packed bits for one color of one output line does not exceed 16383 bytes. If the image width is too large the feature "multiple data sources" is deactivated automatically.

5.3 Passthrough of JPEG-files

Previous version of bmeps always used the jpeglib library to read and decode JPEG images and used run-length, flate and ASCII85/ASCII-Hex encoding for output. This takes time, output files are much larger than the original JPEG files.

When processing JPEG input files bmeps 2.0.0 checks whether or not a 1:1 pass-through of DCT encoded data is possible.
In this case only a fast ASCII85/ASCII-Hex encoding is applied.

5.4 Flexible number of bits per component

Previous versions of bmeps only produced 8 bits per component output.

Version 2.0.0 can produce 1, 2, 4, 8 and 12 bits per component in PS/EPS output and 1, 2, 4 and 8 bits per component in PDF output. If the input bpc value does not match one of these numbers, bmeps attempts to use the next larger value or restricts output to 12 or 8 bpc.

5.5 PDF output

Previous bmeps versions only produced PS/EPS, version 2.0.0 also produces PDF.

For PNG-to-PDF conversion there already was png2pdf, but bmeps 2.0.0 can also process JPEG, NetPBM and partially TIFF.